GAME DESIGN & DEVELOPMENT: USING COMPUTER GAMES AS CREATIVE AND CHALLENGING ASSIGNMENTS

By

CHERYL SEALS* JACQUELINE HUNDLEY** LACEY STRANGE MONTGOMERY***

** Assistant Professor, Department of Computer Science and Software Engineering, Auburn University.*
*** Ph.D Student, Department of Computer Science and Software Engineering, Auburn University*
**** Ph.D student, Auburn University*

ABSTRACT

This paper describes a game design and development course. The rationale for forming this class was to use student excitement with video games as an intrinsic motivation over traditional courses. Today's students have grown up exposed to gaming, interactive environments, and vivid 3D. Computer gaming has the capacity to attract many new students to computer science and information technology majors. The rationale of creating a set of game design classes utilizes gaming as a teaching tool to attract and instruct students with familiar methods and environments. This work will introduce the development of an introductory game design class, its structure, artifacts created and student and instructor's reflections.

INTRODUCTION

In recent years there has been a lot of excitement about computer gaming as evidenced by the increase in video game sale by the computer gaming industry. In April 2007, there was a 15% increase in game sales (Brightman, 2006). With this increase in interest, the authors wanted to leverage this enthusiasm and have a carry over effect into the *classroom*. Computer gaming will also increase student motivation and attract students to the computer science major. The authors wanted to take advantage of intrinsic motivation. "Intrinsic motivation, also known as self-motivation, refers to influences that originate from within a person which cause a person to act or learn" (Bomia et al., 1997). Today's students have grown up exposed to gaming, interactive environments, and vivid 3D. The rationale of creating a set of game design classes utilizes gaming as a teaching tool to attract and instruct students with familiar methods and environments. This work will introduce the development of an introductory game design class, its structure, artifacts created, and student and instructor's reflections (Brightman, 2006).

Computer Science departments worldwide and across the nation have seen a drastic decline in student enrollment. This is evidence of a growing problem that we will not have enough students in our majors to fill the great need for technology careers. The Monthly Labor Review specifies that by 2012, the United States will have 500,000 technology jobs that we will not have the personnel to fill in this country. How to get students more excited about technology in general and specifically computer technology and programming?

This class was created with interdisciplinary student population in mind. There was detailed introduction to computer gaming theory, design and development, with an introduction to the gaming industry. Does this interdisciplinary type of class fit into the traditional set of classes within the CS major? This would be very challenging because of accreditation, but would fit nicely as a special topics class or as a class leading to a game design certificate. With so many courses that are required (e.g. our department only allows 2 electives) it would be very challenging for students to take this course unless highly motivated by the subject matter. So if students take Game Design and Development, they must currently take it as an independent study and not as a

component of the standard curriculum.

Background

Teaching Software Engineering and programming through a game design course can be highly beneficial to students in that it integrates a wide array of topics which is more realistic and in pragmatic fashion. In order to create a computer game, a student would need a toolkit of methods. They would learn these methods from prior experience and new information presented in a gaming course. This type of development effort would combine the artificial intelligence, software engineering, graphics, networking, and human computer interaction (Claypool & Claypool, 2005). The application of skills that are highly transferable is always to the student's advantage. This will provide a practical, yet fun alternative to homework reinforcing breadth and depth of knowledge (e.g. computer programming and application of music and art knowledge).

Game Design and Programming will utilize fundamental computer science subjects in practical ways and will incorporate concepts from physics, mathematical modeling, and game specific principles that are not generally covered with rigor in traditional computer science curricula. In an attempt to provide enrichment and classes for the non traditional student many game trade schools have worked to capitalize on the burgeoning interest in computer gaming. They have offered many game classes, but the game industry decries this approach. They need students who have a full foundation in computer science with exposure to networking, programming proficiency, algorithms, and artificial intelligence. The gaming industry require students trained not only in the areas that support programming, but well rounded students who are also trained in liberal arts (e.g. history, art, music, creative writing). This gives the students a firm foundation and develops accomplished scholars and game designers (Coleman, Krembs, Labouseur & Weir, 2005). Basic programming can be taught, but creativity is not something that one just mass produce on an assembly line. It is required to develop non linear thinkers to create something truly innovative.

"We found that it might be more effective to teach the art design students how to program, rather than to teach the CS student how to draw" (Tsai, Huang, & Zeng, 2006). This might prove difficult, but many of game programming activities are scaffold by the use of game programming tools with graphical user interfaces (e.g. 3DGameStudio™, Virtools™, Quest3D™, HalfLife2, etc.)"

In addition, by learning to program the art/design students improve their technical communication skills, which allow them to better interface with the programmers in the future. "This is a skill which is necessary in the game industry, where programmers and art designers must work very closely together to get their work done" (Tsai, Huang, & Zeng, 2006).

One of the biggest determining factors of student success is motivation, proper foundation, and determination. The hope in designing this course is to take advantage of the powerful lure of gaming. Many students are motivated by the desire to be a part of shaping the future games industry. The rationale was to engage student interest, and to develop a course to attract students to computer science and computer technology.

Why include gaming courses in computer science curricula?

There are various reasons to provide gaming courses in the computer science curriculum. Intrinsic Motivation is good reason to create game courses. "Motivation is student's willingness, need, desire and compulsion to participate in, and be successful in the learning process" (Bomia et al., 1997). With the help of intrinsic motivation, instructors have enthusiastic harder working classes, while extrinsic motivation like grades in many cases is not as compelling for many students. Gaming courses will take advantage of intrinsic motivation to support recruitment and retention efforts, because many students are interested in games and can leverage the enthusiasm for gaming in the classroom. Many freshmen are interested in computer science because of their love of video gaming and the opportunity to create the things they love and spend enormous amounts of time with. Gaming

courses can provide real world experience in research based learning and game programming.

The rationale for the course design was to attract students who were interested in game playing. In many cases with future instantiations of the class a true interdisciplinary mix would be appropriate. In most gaming companies there are groups that would work with story & plot, graphics, animation, music, networking, etc. To have a truly industrial strength game experience, the game development team members need to become mini-experts in areas that support game development.

The idea of Game Development team throws an extra caveat into the class. The projects are software engineering projects where team members must collaborate and share ideas to have successful projects (e.g. pair programming). The hope with pair programming exercises is that members don't get stuck as often and that they can feed off one another's enthusiasm and knowledge. This is more practical and realistic, because in many real world projects programming is not done in isolation for all of your work. Members are accountable and their work must be integrated into the whole. Also with just one semester there was not enough time to be exposed to the richness of gaming design and development. There are just not enough curriculum hours and many problems and frustrations were caused by not having a dedicated gaming lab.

Game Class Structure

The question which came to mind, "How do we improve computer science & software engineering enrollments?" The investigation began by jumping on the information superhighway to search for the Holy Grail to hold the attention of teen and young students. In conversing with other instructors, colleagues, friends and family, one thing that kept staying in the fore front of the conversations, was video games and the gaming industry and that "you can't get students to their homework, but you can't get them to put down their video games". This fueled the idea that a game development class should be started as an elective to gauge student interest to study game theory and game development at near the same levels they are enamored with video games.

Background research also continued by talking to colleagues at the first game design conference to find out the current state of the art in game design and in game course material. From that meeting, a myriad of useful things to investigate (game industry leaders, instructors, their websites, links to course pages, books and other materials) were found. With this ammunition, two courses "Introduction and Game Design" and "Game Design and Development" were designed. The book chosen for the course is a unique book in that it combines the wisdom and expertise of over 25 game industry professionals to give an unprecedented view of game development from game design to programming, to production and business issues (Rabin, 2005). A secondary text "Game Design Workshop" was used to direct more application oriented section of the class (Fullerton, Swain & Hoffman, 2004).

The game design text was the framework for the game theory section of the class and was based on the curriculum framework proposed by the International Game Developers Association (IGDA). The Game class structure had three major components which are theory, application and presentation.

Component one is game theory with lecture, assignments and illustration, Component two is application of game theory by creating a game during the class that can be played and is fun., Component three is the presentation of special game topics to give students more ownership of the content material and the intermediate and final game presentation included a show and tell of their final creations.

Component One: Game Class Theory

The Rabin text, presented three sample curriculum models to focus course structure: programming-oriented, inter-disciplinary and game design. The inter-disciplinary with the rationale of the broadest coverage of topics was chosen to support students who may be interested in gaming, but who may not have an extensive background in programming and

may be more interested in game design. The class began with an introduction, overview of video games and some of the other topics are as follows:

Readings in critical game studies: Ludology for game Developers

- The Games Industry
- Readings in Understanding Fun
- Finding Flow
- Game production & teams
- Intro to gaming Application (3Dgame studio)
- Game Design Documentation
- Introduction to Level Design
- Modeling
- Modeling Application
- Human-computer interaction (HCI)
- Game Development Project Work
- Interface design Computer
- Computer Graphics
- Game scripting and programming
- Collision Detection Extras
- Game data structures and algorithms
- Artificial intelligence
- Network Extras
- How to Prototype a Game in Under 7 Day
- Play testing (Game Usability Testing)

Facilities

With this being an experimental class, there were problems based on obtaining adequate facilities and software support for the class. With space at a premium we were able to place some equipment and software in an educational lab, but this was problematic in that everything is recycled each semester which caused the loss of some software and other tools. For the Game Design and Development class, students relied on individual downloads from free game design engines which caused consistency problems. For future iteractions of the class, there is necessity of having dedicated lab facilities for an application and development driven class.

Component Two: Game Design Projects

The rationale for the game design course was to create a class to motivate students to explore other types of software not traditionally used in Computer Science classrooms. The projects were all designed to be educational software that would be supportive to middle and high school lesson plans. An initial engine was suggested for the first game programming effort, but many found frustration in this environment, in that it was on the bleeding edge of development with frequent updates and documentation (e.g. tutorial support) of iterative development, which where levels behind recently released software. This inconsistency caused half of the groups to choose other software for their development work.

All student teams were encouraged to explore and find software that would best support their project goals of developing creative educational software. This was an interesting way to approach the topic and yielded many different types of products. But not mandating one environment was problematic because the instructor only had experience with one environment, which meant that in-depth level support would not be available beyond the precursory level of mechanics. This introduces diverse software usage issues, which encourage exploration to find support materials. Also each team needed to investigate the appropriate means in the chosen environment to complete the game programming project in successful and effective ways.

The application component of the class was an opportunity for students to practice their software engineering skills and go through the entire software engineering lifecycle. They had to create requirements, create initial design, prototype games and play test games to receive feedback about their designs. In all there were six games created in the Game Design and Development class. In the 3D game category there were three games (engine): Xtreme Rally, Mindcept and PsychGame. Xtreme Rally is a car racing game created with 3D game studio (Figure 1). Mindcept was created

with Half-Life 2 source engine as an adventure game and players solve puzzles to advance (Figure 2). PsychGame was created with 3D game studio engine and has a series of exercises based upon Pavlovian conditioning (Figure 3).

Also in the 2D category there were three games: Destination Travelers, Ditto, and Street Legal Customs. Destination Travelers was created in Flash and the game depicts a family on vacation and the choices they are presented with. Ditto was created with the jMonkey engine as a series of puzzles for youth to teach recognition through matching drills (e.g. mathematics facts). "Street Legal Customs" was created in Flash as a game for youth financial management and the players buy parts for cars based on their credit. All of the games created during the class were very interesting, playable and well liked by their classmates. As an illustration of the breadth of experience needed to create a functional game, an example of one of the games created during the course is presented.

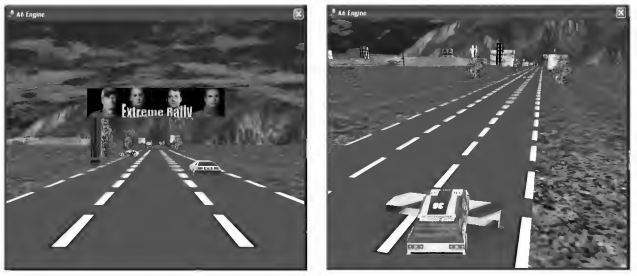


Figure 1. Extreme Rally Car Race.

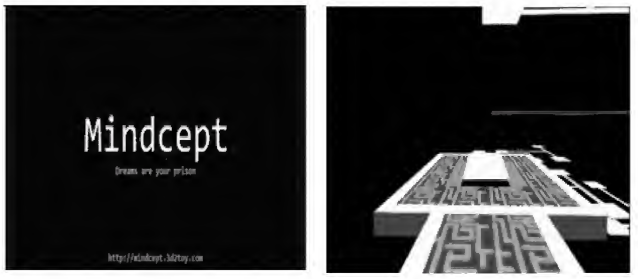


Figure 2. Mindcept: Dreams are your prison.

Figure 3. PsychGame.

DITTO: An educational game

Ditto was one of the games created for the game design and development class. Ditto is an educational puzzle game with mechanics similar to games like Concentration and Memory where the player tries to find matching pairs. Each pair consists of related information, like two with the same color or one with a term and the other with information about the term.

Figure 4 shows an example where two matching pairs have been found. These four blocks have been removed revealing four squares of the background picture. The user, then, selects two more blocks, the word "Duck" and the elephant picture. These blocks do not match and will be turned back around to the side containing the letter. Figure 5 also demonstrates Ditto as a multi-player game. At the bottom of the screen, the scores of two players are visible.

Ditto fits the educational and puzzle genres. Education games teach as they entertain. Puzzles are categorized as casual games which combine pattern matching,

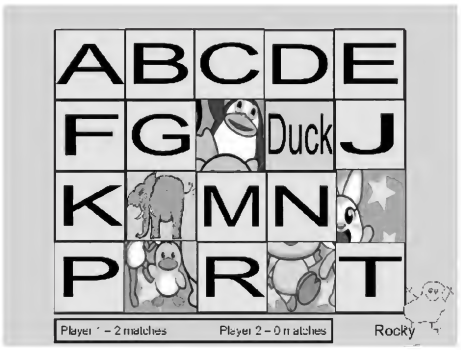


Figure 4. Animal names

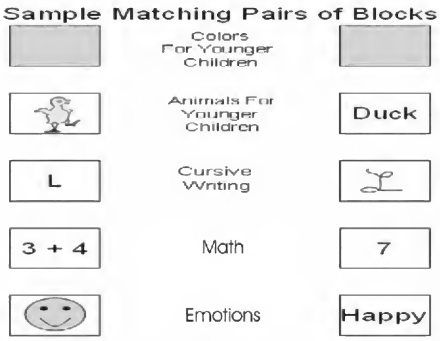


Figure 5. Sample Matching Pairs of Blocks.

logic, strategy, luck, and often time elements. They do not have a story as common adventure game, but are an end themselves. The goal in designing a puzzle game is not to make the player feel stupid, but to allow him to challenge himself and to help him win. Most puzzles are un-timed and allow trial and error without penalty. These characteristics make puzzles a good educational medium. Educational game developers must work closely with experts in the field of knowledge presented in the game. There should be a clear goal of what the player will learn through the game. The game should present age appropriate information in an age appropriate manner. The interface should be designed in such a way not to clutter the screen with confusing objects. Every time a player does something, he wants to see something happen. Educational games should encourage the player to continue. This can be accomplished by earning points or receiving encouraging responses to the play action. It is important to address inactivity in educational games (Bates, 2004).

Game Development Strategy

The goal for the development of the game Ditto was to implement multiple categories (Figure 5) and levels of difficulty from which the user can choose. The categories included Chinese Symbols, Math Facts (addition, subtraction, multiplication, and division), sign language, shapes, colors, etc. The levels of difficulty (easy, medium, and hard) determined the numbers of blocks in the game puzzle.

The first layer simply implemented one category, animal names, at the lowest level of difficulty. After this scenario was playable, subsequent layers were developed allowing the selection of a category and level of difficulty using a menu. The information about each game category was stored in a set of XML files. The game engine had a game state manager to allow for intuitive management of game states.

Game Engine

The Java Monkey Engine (jME) was chosen as the basis for the game. The jMonkey Engine (jME) is an open source from (jMonkey, 2007) with high performance scene graph based graphics API. It was built to provide a full-featured graphics engine written in Java. It allows any rendering system to be plugged in by using an abstractor layer. jME supplies the user with easy to use, but powerful classes for building the application.

At a fundamental level, it uses scene graph architecture. Each game state constructs a tree to be rendered by the engine. This allows for a hierarchical scene design to be translated directly into code. The texturing capabilities of the engine were used and found to be fairly intuitive. No attempt was made to load any externally built models or animations. As for other libraries, FengGUI was used to create the 2D menu interface. FengGUI is an openGL GUI toolkit. JME allows for the use of both Swing and AWT, and it is likely that other toolkits could be integrated. For xml parsing, a library called dom4J was used which is a wrapper around javax.xml. It is quite convenient for most XML processing purposes.

Game Development Teams

In Game Design & Development class, students were instructed that interdisciplinary team can be highly productive and have great outcomes. A key to success in game development is a development team of diversified skills. There are many team components when developing a game such as storyboarding, graphics design, Artificial Intelligence, Audio, Dialog, Technical Writers, etc. Although many of the computer science students developed games, which revolved around computer programming at the end of their development process they began to see the benefits of having other components like formally trained technical writers to work on dialog or story, artists to work with animation, musicians to work with audio design, etc. There are many other aspects that are highly important to a good compelling game.

The first task was to designate workable team structures and all groups were allowed to form based on a group synergy. Two groups were single member teams (e.g. PsychGame and MindCept). One group consisted of three team members (i.e. Ditto) with one developer, one graphic designer, and a technical writer. This group had

good synergy and their game was successful and was well received by the class during playtesting. Two groups had four members (i.e. Xtreme Rally and StreetLegalCustoms). In both groups, majority of the members were assigned to program different components of the games and one technical writer to organize writing, internal group testing, external playtesting and presentations. This team organizational structure worked very well for these two teams. The whole team worked together to define the vision for their game, create a name for their project group, and started thinking about marketing, in case their game develops a liking/following. The first class milestone was complete when groups presented their game vision and the tentative schedule for management of people and time resources.

The second step for each team was to develop a website to help manage group resources and as a dissemination point for project deliverables. This website contained project abstract, description, presentation, and in some cases the actual game or video. Team Ditto created a very personable website that was very reflective of the group's synergy for their game design project.

The next step was to design and develop interesting, fun and playable games. The Game Design & Development group all conducted iterative development and rapid prototyping. They began with Designing their game, creating specifications, coding and testing, which were fed back into iterative redesign of the game until alpha release of their games were ready for class playtesting.

The final step in the rapid design cycle was unit testing and bug testing to get the applications working properly and preparing for playtesting of their created works by other students from their class. In all cases, players found that the appearance, user interface and game interaction were very important. In team Ditto, all of their team members were computer science majors, and relied on the coding strength of one member and the design skills of the others.

It was found that working in game development teams was highly instructive for all students in that they learned how to compromise, be flexible, work within a team and realized that all members of a team will have unique contributions. The teams, in many cases, had varying levels of ability and skills in each area, which in many cases proved challenging and frustrating. Even with introductory materials to train the students in the benefits of pair programming and working in teams, the semester was concluded with two one creator games. Each of these designers wanted to render their artistic vision without compromise and in the end developed a solo effort. These individuals were very skilled programmers and were able to create working games during the project period of the class time utilizing game engines. Each game was designed, developed, unit tested and playtested in a 4-8 weeks period based upon the size of team and size of the final game.

Component Three: Game Presentations

The rationale for game presentations was twofold, to serve as milestones for deliverables and to reinforce student professional development. Also in many cases students will have to give professional presentations in the future either in the academic world as teachers or in the business world as training sessions they provide for their colleagues and trainees. Each group gave three presentations during the semester. The first group presentation described their game ideas with the theme, intended game play with paper prototypes and sketches of proposed setting. The second presentation for each group was a midterm status report to give all an indication of their progress and as a reminder that their time budget for rapid prototyping was half over. The final presentation for each group was their final presentation with the final game status, results of playtesting, and demonstration of the working game.

Results and Future Work

The results for the instructor were, that in the future they will continue with more offerings of the game design and development class. Many students were frustrated by the theoretical portions of the class, but were highly motivated by the class game projects. Many were motivated by wanting to have a great game that their

peers and they would enjoy. The results from the Ditto team were, that they were very proud of the cleanness of the code. The game action was simple but challenging. Ditto can be marketed as a learning reinforcement tool. They were motivated by their experience and thought that given more time, they would expand the game by adding more categories. Other features that would enhance the game are: a high score list, keyboard block selection, two players, music and sound, and online play.

Conclusion

From the instructor perspective, the class met its intended baseline of introducing students to the gaming industry, game design and development, and the creation of educational games. There were many varying levels of students' game attainment based upon overall effort and coordination of groups. Many suffered from not understanding pair programming and group projects, knowing that they have to work together from the beginning to make sure that the project meshes well during the end of a project. For the groups that understood that there is a division of labor and coordination that is necessary for a successful programming project they received much better results in the end with much less trauma. Groups that waited until late in the game to coordinate their resources in many cases could not or had to do some recreation in order for their games to work successfully. Our assumption is that students performed higher based on being more motivated to bring their own creations to life virtually and that a game enhanced course will motivate students better than traditional instruction.

From a student's perspective, a gaming course offered the practical application of the software engineering and human computer interaction skills they have learned in traditional Computer science courses. The projects allowed for the integration of many forms of creativity with respect to computer science skills. Investigating types of games and interface possibilities was interesting and the designing of the interfaces and interactions was enjoyable. Most teams possessed a wide array of coding, graphic design and interface design skills that they drew upon to complete their projects. The students also expressed that working together was a rewarding activity during the class.

References

[1]. Bates, B. (2004). *Game Design* (2nd ed.). Boston: Thompson Course Technology.

[2]. Birdwell, K. (1999). *The Cabal: Valve's Design Process For Creating Half-Life*. Retrieved November 16, 2008 from http://www.gamasutra.com/features/19991210/birdwell_01.htm.

[3]. Bomia, L., Beluzo, L., Demeester, D., Elander, K., Johnson, M., & Sheldon, B. (1997). *The impact of teaching strategies on intrinsic motivation*. Champaign, IL: ERIC Clearinghouse on Elementary and Early Childhood Education. (ERIC Document Reproduction Service No. ED 418 925).

[4]. Brightman, J. (2006). U.S. Video Game Sales Up 15.5% in April. *GameDaily Biz*. (May 16, 2006). Retrieved October 8, 2007 from http://biz.gamedaily.com/industry/feature/?id=12700&rp=49.

[5]. Claypool, K. and Claypool, M. (2005). Teaching software engineering through game design. *In Proceedings of the 10th Annual SIGCSE Conference on innovation and Technology in Computer Science Education* (Caparica, Portugal, June 27-29, 2005, ITiCSE '05) ACM Press, New York, NY, 123-127.

[6]. Coleman, R., Krembs, M., Labouseur, A., and Weir, J. (2005). Game design & programming concentration within the computer science curriculum. *In Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education* (St. Louis, Missouri, USA, February 23-27, 2005). SIGCSE '05. ACM Press, New York, NY, 545-550.

[7]. Fullerton, T., Swain, C. & Hoffman, S. (2004). *Game Design Workshop: Designing, Prototyping and Playtesting Games*. San Francisco, CA: CMP Books.

[8]. jMonkey Engine. n.d. Retrieved March 8, 2007 from www.jmonkeyengine.com.

[9]. Lewis, M. and Jacobson, J. (2002). Game engines in scientific research. *Communication of the ACM* 45, 1 (Jan. 2002), 27-31.

[10]. Rabin, S. (2005). *Introduction to game*

development. Hingham, MA: Charles River Media, Inc.

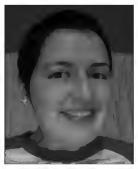
[11]. Smith, M. (2008). *V Game equips first responders to save lives: New virtual world hones paramedics' triage skills*. Retrieved July 5, 2008 from http://videogames.Yahoo.com/feature/game-equips-first-responders-to-save-lives/1223748.

[12]. Tsai, M., Huang, C., and Zeng, J. (2006). Game programming courses for non programmers. In *Proceedings of the 2006 International Conference on Game Research and Development* (Perth, Australia, December 04-06, 2006). *ACM International Conference Proceeding Series,* vol. 223. Murdoch University, Murdoch University, Australia, 219-223.

ABOUT THE AUTHORS

Cheryl D. Seals, Ph. D. is an Assistant Professor in the Computer Science and Software Engineering Department at Auburn University. She conducts research in Human Computer Interaction with an emphasis in visual programming of educational simulations with end user programming, intelligent agent, usability evaluation, computer supported collaborative work, minimalism and additionally she is also involved in software engineering projects.

Jacqueline H. Hundley is a Ph.D student in the Computer Science and Software Engineering Department at Auburn University. She conducts research in Software Engineering with an emphasis in curriculum development and software engineering tools usage for CS1/CS2.



Lacey S. Montgomery is a Ph.D student at Auburn University. She graduated Magna Cum Laude with a Bachelor of Software Engineering degree from Auburn University in Fall 2004. Lacey is president of the local Upsilon Pi Epsilon honor society. Her research is in the area of sensor networks and location sensing and visualization. Lacey currently teaches Java programming lab.